Java Web

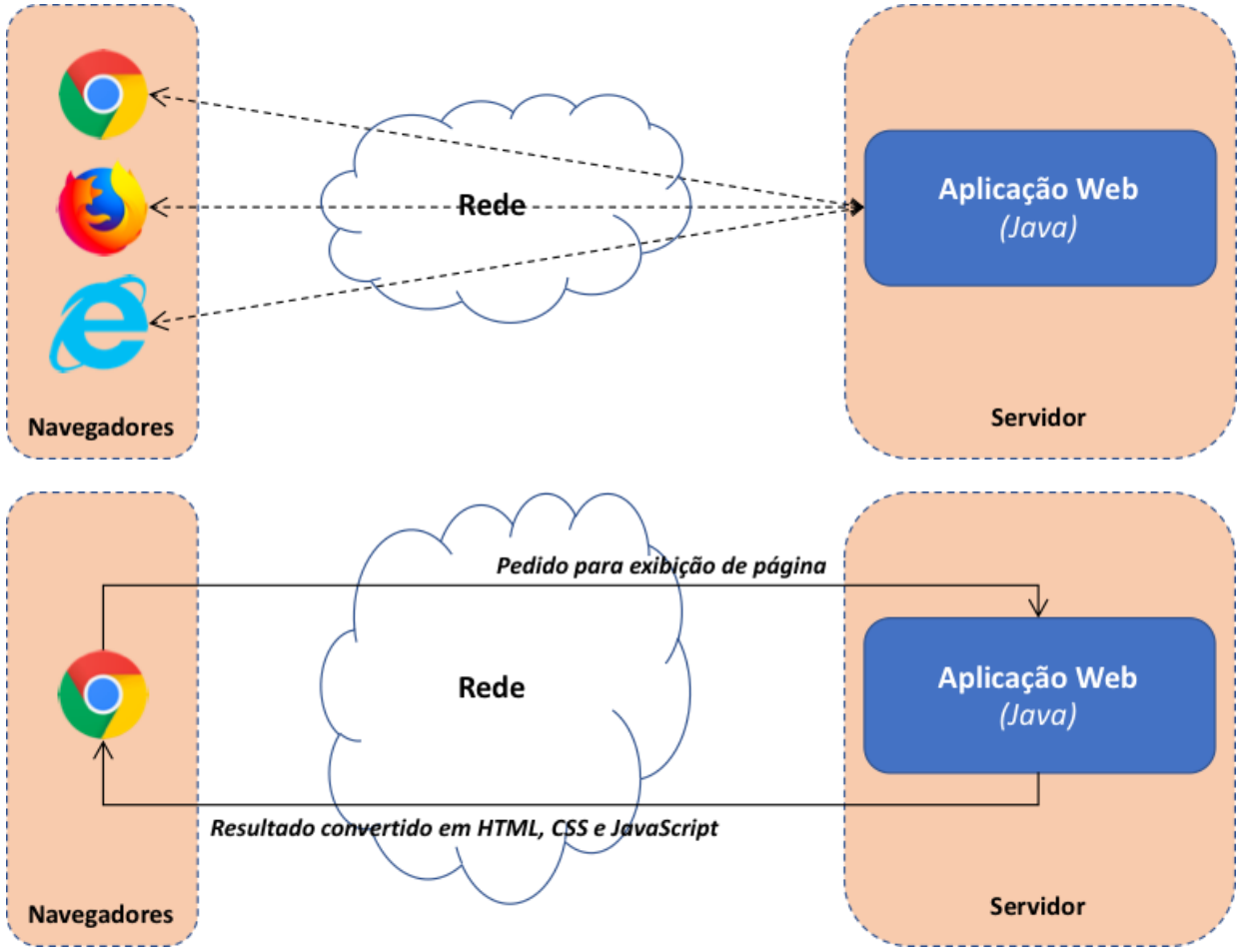
RECODE pro

O ambiente típico de aplicações web

Aplicações web funcionam de maneira diferente de aplicações desktop, enquanto aplicações desktops são instaladas direto nas máquinas locais os sistemas web depende de servidores e tecnologias de front e back end para funcionar de forma correta, Quando falamos do front-end, geralmente estamos falando das estruturas que são responsáveis pela maneira como a aplicação será exibida para o usuário. O front-end é fundamentado em três tecnologias principais: HTML, CSS e JavaScript.

O back-end pode ser escrito em qualquer linguagem que dê o suporte para o desenvolvimento de aplicações web, como é o caso justamente do Java com o JEE. Com o browser, nós acessamos a aplicação através de um *link* que aponta para um servidor. O servidor é onde nossa aplicação web fica instalada e é disponibilizada para os diferentes usuários justamente através desse *link* de acesso.

O ambiente típico de aplicações web



Request: As requisições do protocolo HTTP são basicamente linhas de texto que identificam o "método" da requisição, o endereço do recurso desejado e eventuais informações adicionais a respeito do cliente e das suas capacidades.

Response: As respostas HTTP representam o conteúdo que o atendimento da requisição previamente enviada pelo cliente.

O ambiente típico de aplicações web

O protocolo HTTP (*HyperText Transfer Protocol*, ou Protocolo de Transferência de HiperTexto**)**

Aplicações web modernas rodam em cima de um protocolo de rede que provavelmente você já ouviu falar: o protocolo HTTP. Para que consigamos desenvolver aplicações web de maneira correta, precisamos, antes de qualquer coisa, entender os princípios básicos que regem o protocolo HTTP.

Método	**Significado semântico**
GET	Significa que queremos “pegar” algo no servidor: uma página, por exemplo. Requisições GET fazem com que o servidor devolva algo para o cliente, algo que estava “dentro” do servidor
POST	Significa que estamos querendo incluir alguma coisa no servidor. Por exemplo, se temos uma página de cadastro de usuários, a requisição que vai fazer com que o servidor faça o insert no banco de dados deve ser uma requisição POST, afinal, estamos criando um novo item que vai ficar no servidor
PUT	Significa que estamos querendo atualizar alguma coisa no servidor
DELETE	Significa que estamos querendo apagar alguma coisa do servidor

O ambiente típico de aplicações web

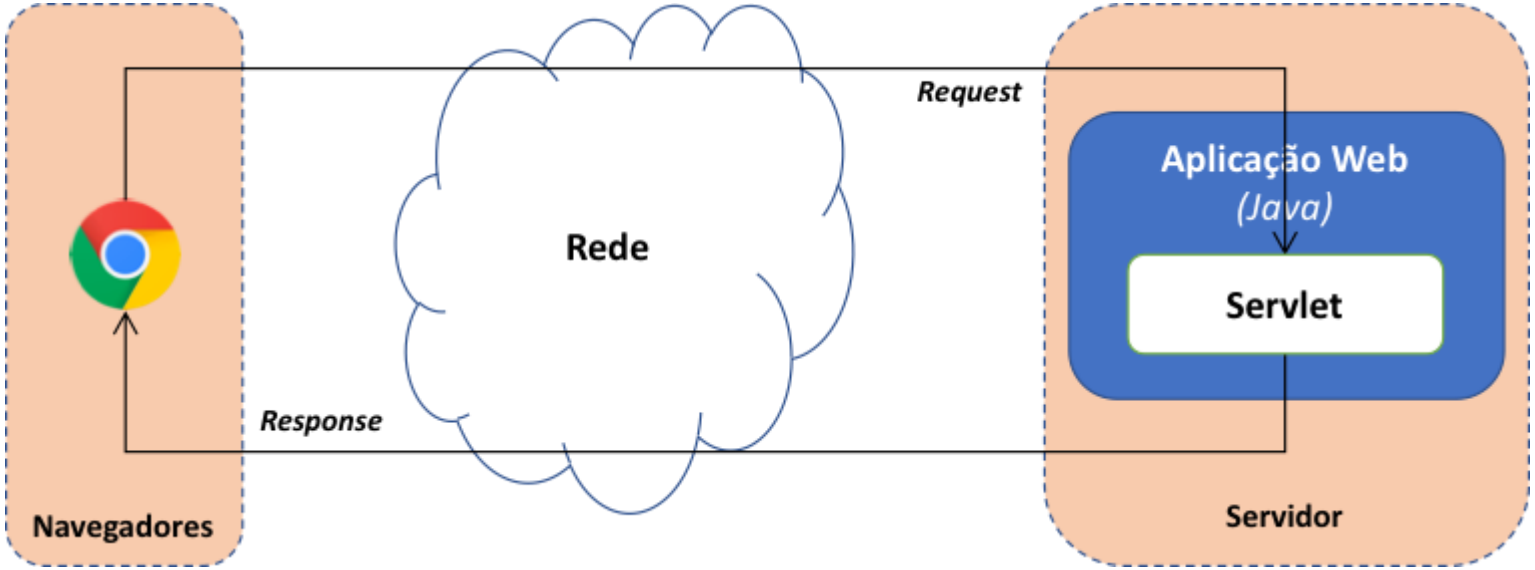
O protocolo HTTP

Os status HTTP também são padronizados pela especificação. Os principais status HTTP que temos são:

Status	Descrição
200	OK. Significa que o servidor entendeu a requisição e a processou sem problemas.
302	Found. Significa que o recurso solicitado de fato existe no servidor (status típico de requisições GET)
401	Unauthorized. Significa que você tentou acessar algum recurso do servidor que exige autenticação para acesso, e você ainda não realizou este processo.
404	Not Found. Significa que você solicitou algum recurso no servidor que não existe no lugar que você indicou. Por exemplo: se você tenta acessar alguma página de algum site que não Existe.
500	Internal Server Error. Significa que o servidor encontrou um erro durante o processamento da requisição.

Elementos básicos do Java para Web: servlets

Quando falamos de aplicações web desenvolvidas em Java, existe um componente que se torna a base da aplicação: o servlet. Por isso, é muito importante que entendamos o que são os servlets e como podemos os utilizar para criarmos nossas aplicações web. Basicamente, servlets são classes que, quando implementadas seguindo-se o padrão definido pelo JEE e são instanciadas em um servidor (como o Tomcat).



Servidor Web Java TOMCAT

O **Tomcat** é um servidor web Java, mais especificamente, um container de *servlets*. O Tomcat implementa, dentre outras de menor relevância, as tecnologias Java Servlet e JavaServer Pages (**JSP**) e **não** é um container Enterprise JavaBeans (**EJB**).

Desenvolvido pela Apache Software Foundation, é distribuído como software livre. Hoje um projeto independente, foi criado dentro do Apache Jakarta.

Fonte: https://pt.wikipedia.org/wiki/Apache_Tomcat

Módulo 04 – Java Web - Aula 01

Tecnologias Web Java EE

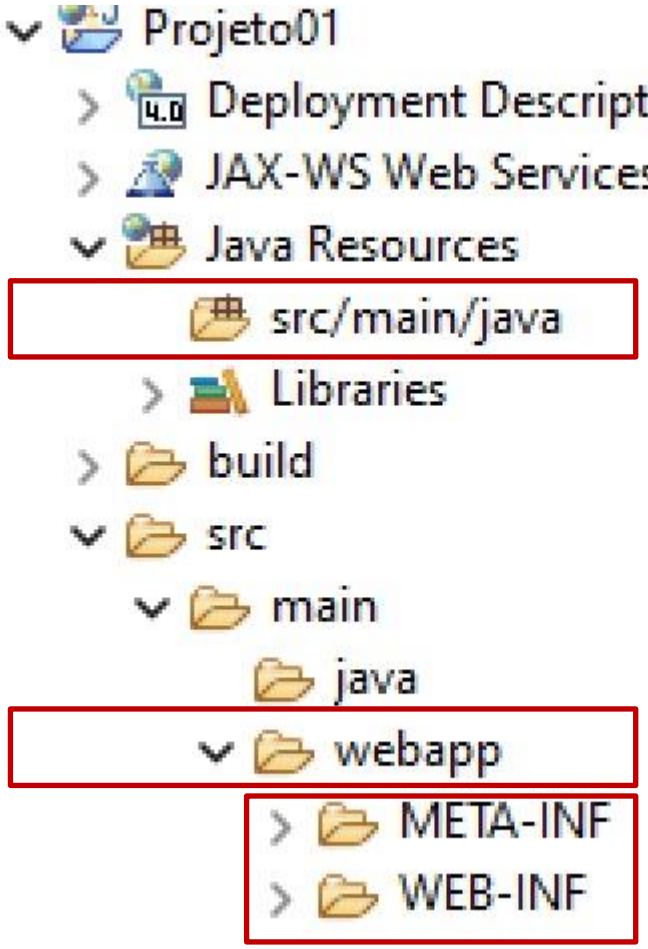
Servlet (servidorzinho em tradução livre) é uma classe Java usada para estender as funcionalidades de um servidor. Apesar dos servlets poderem responder a quaisquer tipos de requisições, eles normalmente são usados para estender as aplicações hospedadas por servidores web

JavaServer Pages (**JSP**) é uma tecnologia que ajuda os desenvolvedores de software a criarem páginas web geradas dinamicamente baseadas em HTML, XML ou outros tipos de documentos. Para implantar e executar JavaServer Pages, um servidor web compatível com um *container* servlet, como Apache Tomcat, Jetty ou Glassfish, é requerido.

JavaServer Faces (**JSF**) é uma especificação Java para a construção de interfaces de usuário baseadas em componentes para aplicações web. Possui um modelo de programação dirigido a eventos, abstraindo os detalhes da manipulação dos eventos e organização dos componentes, permitindo que o programador se concentre na lógica da aplicação.

Módulo 04 – Java Web - Aula 01

Estrutura de pastas e arquivo projeto Java Web



src/main/java: contém o código-fonte Java do projeto.
src/main/webapp: contém os arquivos HTML, CSS e JavaScript da interface do usuário.
META-INF/WEB-INF: contém os arquivos de configuração do projeto, como o arquivo web.xml.

Módulo 04 – Java Web - Aula 01

Exemplo 01 usando JSP (JavaServer Pages)

```
<%@ page language="java" contentType="text/html;
charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>


        <%
                String nome = "Flavio Mota";
                out.print(nome);
        %>


</body>
</html>
```

Nesse primeiro exemplo, nossa página JSP será processada pelo servidor TOMCAT que devolve para o navegador apenas o código HTML, o usuário não terá acesso ao código JAVA

Todo código que estiver entre os delimitadores <% %> são da linguagem Java.

```
<%@ page language="java"
contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
```
esse bloco de código indica que a linguagem usada será Java e configura o tipo de codificação para html e utf-8

Exemplo 02 usando JSP (JavaServer Pages)

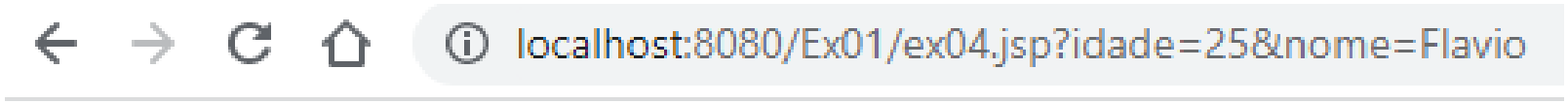
```
<body>
        <%
        int inicio = 1;
        int fim = 20;

        if (inicio < fim) {
                for (int i = inicio; i <= fim; i++) {

                        out.print(i + "<br>");
                }
        } else {
                out.print("O numero inicial deve ser menor que o final: ");
        }
        %>
</body>
```

Exemplo 03 usando JSP (JavaServer Pages)

```
<body>
        <%
        String[] nomes = { "Flavio", "Rochele", "Raquel", "Maxx" };
        %>
        <table border="1">

                <%
                for (int i = 0; i < nomes.length; i++) {
                %>
                <tr>
                        <td><%=nomes[i]%></td>
                </tr>

                <%
                }
                %>

        </table>
</body>
```

Módulo 04 – Java Web - Aula 01

Passando parâmetros via url



Idade enviada via url = 25
nome enviado via url = Flavio

```
<body>
	<%
		String nome = request.getParameter("nome");
		int idade = Integer.parseInt(request.getParameter("idade"));

		out.print("Idade enviada via url = " + idade + "<br>");
		out.print("nome enviado via url = " + nome);

	%>
</body>
```

Módulo 04 – Java Web - Aula 01

Passando parâmetros via link

```
<body>
        <h1>
                <a href="ex05_2.jsp?n1=20&n2=40"> CLIQUE AQUI PARA CALCULAR OS
                        PARAMETROS n1 = 20 e n2 = 40 </a>
        </h1>


</body>
```

Página JSP que faz o request

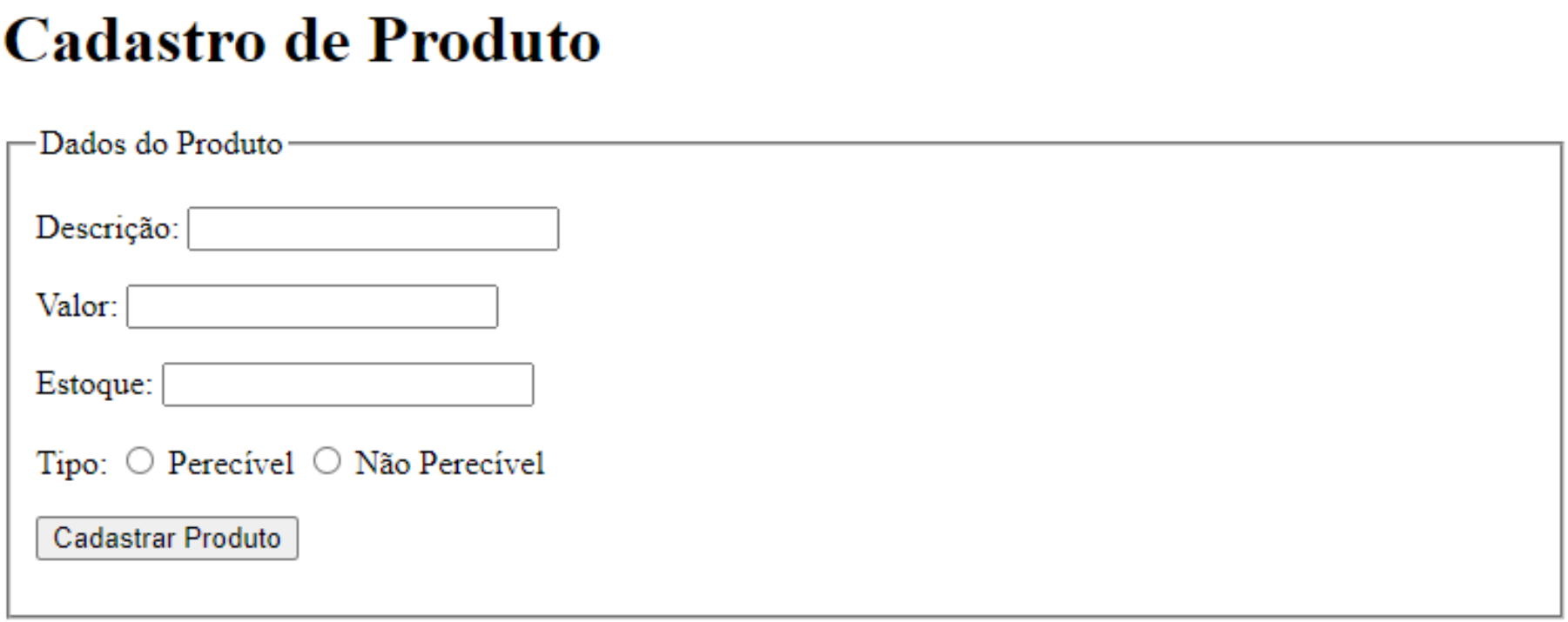
```
<body>
        <%
                        int valor1 = Integer.parseInt(request.getParameter("n1"));
                        int valor2 = Integer.parseInt(request.getParameter("n2"));

                %>

        <h1>
                A soma dos parameteros =
                <%= valor1 + valor2 %>
        </h1>
</body>
```

Página JSP que recebe os parâmetros

Módulo 04 – Java Web - Aula 01

Trabalhando com Formulário

Módulo 04 – Java Web - Aula 01



Cadastro.html

Módulo 04 – Java Web - Aula 01

Cadastro.html

```
<body>
        <h1>Cadastro de Produto</h1>
        <form action="relatorio.jsp" method="post">
                <fieldset>
                        <legend>Dados do Produto</legend>

                        <p><label for="nome">Descrição:</label>
                        <input type="text" id="nome" name="descricao"></p>

                        <p><label for="idade">Valor:</label>
                        <input type="number" id="idade" name="valor"></p>

                        <p><label for="peso">Estoque:</label>
                        <input type="text" id="peso" name="estoque"></p>

                        <p>Tipo:
                        <input type="radio" name="tipo" value="perecíveis ">
                        <label for="canina">Perecível </label>
                        <input type="radio" name="tipo" value="Nao_perecíveis ">
                        <label for="felina">Não Perecível</label></p>

                        <p><input type="submit" value="Cadastrar Produto">
                </fieldset>
        </form>
</body>
```

Módulo 04 – Java Web - Aula 01

```
<body>
        <%
        String descricao = request.getParameter("descricao");
        float valor  = Float.parseFloat(request.getParameter("valor"));
        int estoque  = Integer.parseInt(request.getParameter("estoque"));
        String tipo  = request.getParameter("tipo");
        %>

        <h1>Cadastro de Produto</h1>

        <p>O seguinte produto foi cadastrado com sucesso:</p>

        <p>
        Descrição:<%=descricao %><br>
        Valor: <%=valor %><br>
        Estoque: <%=estoque %><br>
        Tipo:<%=tipo %>
        </p>
</body>
```

Relatorio.JSP

Módulo 04 – Java Web - Aula 01

Exercício 01

Calculadora.html

Criar uma página JSP para processar os cálculos e mostrar o resultado

RECODE

www.recode.org.br

RECODE pro

recodepro.org.br

Institucional

/rederecode

/recoderede